Defect Tolerance in Hybrid nano/CMOS Architecture using Tagging Mechanism

Saket Srivastava, Aissa Melouki and Bashir M. Al-Hashimi
School of Electronics and Computer Science, University of Southampton, UK
(ss3, am06r, bmah)@ecs.soton.ac.uk

***Abstract*—In this paper we propose two efficient repair techniques for hybrid nano/CMOS architecture to provide high level of defect tolerance at a modest cost. We have applied the proposed techniques to a lookup table(LUT) based Boolean logic approach. The proposed repair techniques are efficient in utilization of spare units and viable for various Boolean logic implementations. We show that the proposed techniques are capable of handling upto 20% defect ratess in hybrid nano/CMOS architecture and upto 14% defect rates for large ISCAS'85 benchmark circuits synthesized into smaller sized LUTs.**

I. Introduction

Hybrid nano/CMOS architecture has shown promise in bridging the gap between CMOS and emerging technologies [1]. Lithography-based technology used in current CMOS fabrication and bottom-up fabrication approach using self assembly has so far proven to be inadequate in building reliable nano circuits. However, the tremendous gain in device density that can be achieved using nanoscale systems presents a compelling case for developing hybrid nano/CMOS computing architecture [2], [3], [4] where unreliable but highly dense nano/molecular systems are used to provide data storage and computation while CMOS components are utilized for interfacing and for highly critical circuit operations. To achieve acceptable levels of defect tolerance for nano/CMOS architecture efficient repair techniques need to be implemented.

The defect tolerance capability of various repair techniques can vary greatly for different architecture designs. Recently, an efficient repair technique called Repair Most [5] has been proposed to target terabit scale memories using hybrid nano/CMOS architecture. Our analysis of the Repair Most technique when applied to LUT based nano/CMOS computation architecture shows that the technique is unable to handle high defect rates (as seen in section III). While exact manufacturing defect rate is not yet pinpointed, it is believed to exceed 10% [6]. Hence there is a need for more efficient repair techniques that target higher defect rates. Moreover, most of the earlier works in nano/CMOS co-design have targeted memory [7], [5] or general crossbar architecture [8]. To advance computational nanocircuits, new architectures must be pursued. One such promising architecture is the Look-Up Table (LUT) based Boolean logic architecture considered in this paper.

In this work, we target a hybrid CMOS/nanodevice computational paradigm based on a LUT implementation of Boolean logic functions [9], [10]. We show that the proposed repair techniques are capable of targeting higher defect rates (upto 20%) using a tagging mechanism that results in low CMOS overhead. We demonstrate the repair capability of the proposed techniques on ISCAS'85 benchmark circuits synthesized into smaller LUTs of different sizes. Further, we analyze the efficiency of the proposed techniques in terms of their repair capability and the cost of repair. The novelty of this work lies in the application of previously known memory repair techniques in the context of emerging technology (hybrid nano/CMOS) architecture implemented as LUT. To the best of our knowledge, there are no reported repair techniques that target LUT based architecture implemented in hybrid nano/CMOS technology.

II. Proposed Repair Techniques

In this section we propose two repair techniques that have been developed specifically for LUT based Boolean logic architecture implemented in nano/CMOS. The general repair concept is derived from a CMOS memory repair technique proposed in [11]. However, the technique proposed in [11] is not applicable by itself to LUT based architecture proposed in this work because an individual LUT size is much smaller as compared to a highly dense memory architecture targeted in [11], hence we do not require replacement of blocks of memory unit which is the key difference between our proposed techniques and the memory repair techniques. Moreover, the original architecture if applied to LUT based architecture will impose a significant CMOS area overhead which will nullify the gain in device density achieved by using nano components. Our proposed repair techniques involves replacing rows and columns instead of blocks of defective units. We have also included a *tagging* mechanism to isolate defective rows and columns. Each row/column is associated with a CMOS tag that holds one bit of information. A '1' or '0' tag value specifies whether or not a row/column is selected in the final LUT after repair.

A. Tagged Repair Technique

The aim of our proposed Tagged Repair technique is to identify a defect-free instance of a LUT of size $(2^N \times N)$ within a defective fabric given a certain amount of spare columns c_{sp} and spare rows r_{sp}. Hence, a theoretical estimation of the circuit failure rate of this technique reduces to the calculation of the probability of the non-existance of a subset of defect-free resources $(2^N \times N)$ within the partially-usable fabric

$\big((2^N + r_{sp}) \times (N + c_{sp})\big)$. We first calculate the probability $P_{(col,L)}$ of a column of size $(r+L)$ is defective (i.e. in which the total number of defective bits exceeds the number of spare rows L). The probability of successfully finding enough resources to create an instance of a given LUT using the Tagged Repair technique is: $P_{succ} = \sum_{x=c}^{c+c_{sp}} \binom{c+c_{sp}}{x} P_{inst}(x, r_{sp})$. Where the number of spare rows $L = r_{sp}$ and P_{inst} is the probability that n columns out of $(c+c_{sp})$ are not defective and aligned. $P_{inst} = f(P, c, c_{sp}, r, r_{sp})$ and P is the *defect rate*. Therefore, the overall *failure rate* is given by $P_{failure} = 1 - P_{succ}$.

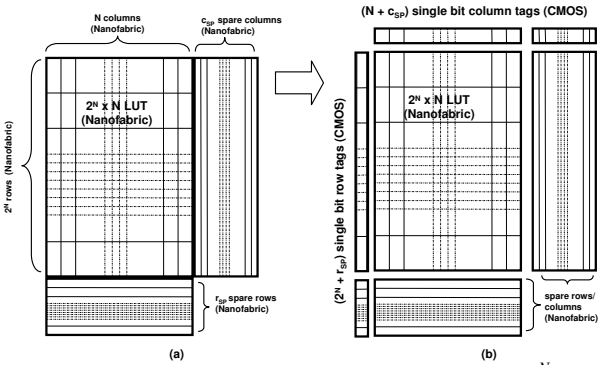

Fig. 1. Tagged Repair Technique: (a) Implementation for a $2^N \times N$ LUT with c_{sp} spare columns and r_{sp} spare rows (b) Use of tags to repair columns and rows

Fig. 1 shows the implementation of the Tagged Repair technique. This technique uses a tagging method to tag rows and columns that are least defective. Initially the tags for original 2^N rows and N columns in LUT are set to 1 and tags for spare rows and spare columns (r_{sp} and c_{sp} respectively) are set to 0. Starting with the original defective LUT, the spare columns are first scanned and then subsequently replaced if less defective columns are found. Similarly, the procedure is repeated for the spare rows. After the repair process, the tags will hold '1' for the least defective rows and columns and '0' for the excluded ones. The proposed architecture is comparatively simpler as it does not require encoding/decoding circuitry that will also lead to additional area overhead in CMOS domain (as compared to other techniques such as [12]). A key advantage of this Tagged Repair technique is that it uses considerably less redundancy to tolerate even higher defect rates compared to Repair Most technique (as we will see in section III). The overall nanodevice area of Tagged Repair technique for a $2^N \times N$ LUT with c_{sp} spare columns and r_{sp} spare rows will be $(2^N + r_{sp}) \times (N + c_{sp}) - (r_{sp} \times c_{sp})$. The CMOS area overhead of this technique is $(2^N + r_{sp})$ single bit row tags and $(N + c_{sp})$ single bit column tags (Fig 1(b)).

B. Modified Tagged Repair Technique

To address even higher defect rates, we investigate a modified technique as presented in Fig. 2 which is a modified implementation of the previous Tagged Repair technique shown in Fig. 1. In this technique instead of replacing entire columns, we have split the columns in two equal sections before

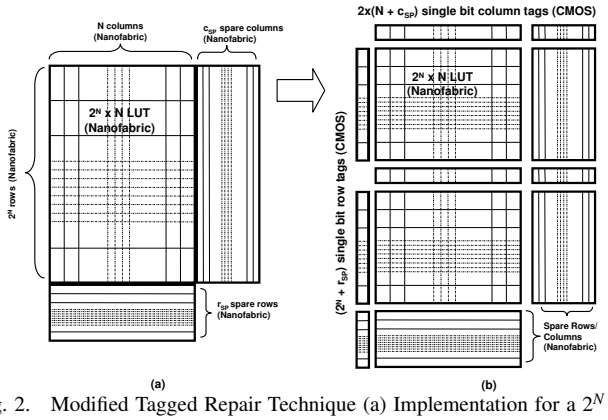

Fig. 2. Modified Tagged Repair Technique (a) Implementation for a $2^N \times N$ LUT with c_{sp} spare columns and r_{sp} spare rows (b) Use of tags to repair columns and rows

applying tagging and replacement, to make more optimized usage of the spare units.

In the Modified Tagged Repair technique, a successful instantiation of a LUT on the fabric is achieved by successfully instantiating each half of the LUT ($2^{N-1} \times N$) on the fabric given the amount of spare columns c_{sp} for each half and the spare rows r_{sp} that is reserved for both of them. Hence for a Modified Tagged Repair technique, $P_{succ} = \sum_{i=0}^{r_{sp}} \left[\left(\sum_{x=c}^{c+c_{sp}} \binom{c+c_{sp}}{x} P_{inst}(x,i) \right) \times \left(\sum_{x'=c}^{c+c_{sp}} \binom{c+c_{sp}}{x'} P_{inst}(x', r_{sp} - i) \right) \right]$. This value of P_{succ} can be used to calculate the failure probability using $P_{failure} = 1 - P_{succ}$. Variable i is the number of spare rows used by our technique to repair the defective rows in the first half, whereas the rest of spare rows $(r_{sp} - i)$ are used in the repair of the second half of the LUT.

In the implementation algorithm for Modified Tagged repair technique, the column-wise scan needs to be done in two stages (as compared to single stage scan in Tagged Repair technique) and the row-wise scan will be done in a single stage. The reason for this is as follows: since a $2^N \times N$ LUT will always have an even number of rows (2^N), it is easy to use this approach in splitting the columns halfway each of size 2^{N-1}. A similar approach to split and tag rows cannot be used since the length of a row can be odd or even depending on value of N and an odd value of N cannot be split in two equal integers. The downside of this approach is that it makes the technique more complex since the number of column tags required will be double that of the Tagged Repair technique (section II-A). This will cause an increase in CMOS area overhead of the tagging circuitry. When compared to the Tagged Repair technique, this technique will require an extra $(N + c_{sp})$ single bit column tags (making it a total of $2 \times (N + c_{sp})$ column tags). Considering a single bit SRAM cell requires 6 transistors [13], the overall CMOS area overhead in terms of transistor count can be calculated accordingly. The number of row tags will be equal to the Tagged Repair technique.

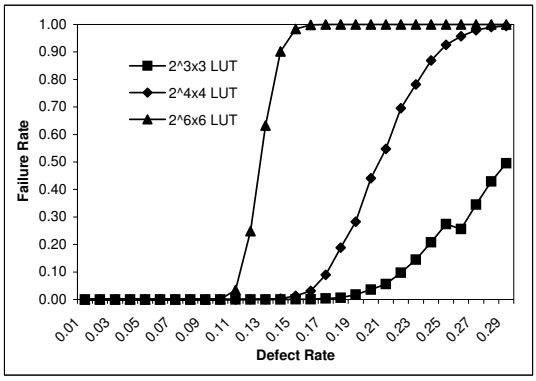

Fig. 3. Plot of Failure rate Vs Defect rate using Tagged Repair technique for different LUT sizes with 100% redundancy in rows and columns.

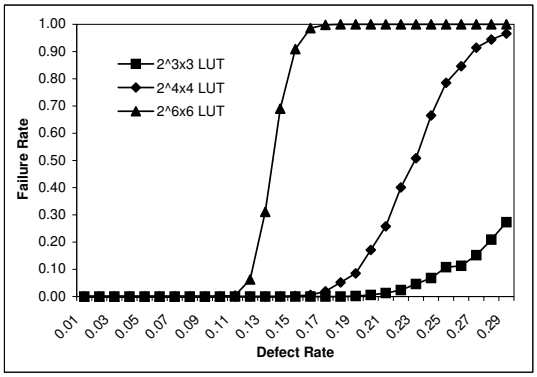

Fig. 4. Plot of Failure rate Vs Defect rate using Modified Tagged Repair technique for different LUT sizes with 100% redundancy in rows and columns.

III. Experimental Results

In this section we first evaluate the performance of the two proposed repair techniques (Tagged Repair and Modified Tagged Repair). Simulations were performed on randomly-generated symmetric LUTs where the probability of 0 and 1 are equal. The LUTs are of sizes ranging from $2^3 \times 3$ to $2^6 \times 6$. Larger circuits (such as ISCAS'85 benchmarks) were synthesized to smaller LUTs using synthesis tools such as Synplicity [14]. The circuit failure probability $P_{failure}$, resulting from randomly injecting m defects, is obtained by calculating the ratio of defective LUTs after repair to the total number of simulation iterations $I = 10000$. *Targeted defect rate* for a particular repair technique is the maximum defect rate for which 0% failure rate can be achieved. *Redundancy (or Spares)* is the percentage of extra rows/columns that are allocated for repair. All the simulations were carried out in C++ and the results were compared with Repair Most technique to determine the gain in repair capability.

A. Simulation Results

Fig. 3 shows the plot of Failure rate Vs Defect rate for different LUT sizes. As can be seen, the proposed repair technique exhibits higher defect tolerance in the case of smaller sized LUTs (such as $2^3 \times 3$ LUT) than larger LUTs (such as $2^6 \times 6$ LUT). Hence synthesis of larger circuits into smaller LUTs will result in improved defect tolerance for the targeted nano/CMOS architecture. The results shown assume 100% redundancy (i.e. $c_{sp} = N$ and $r_{sp} = 2^N$).

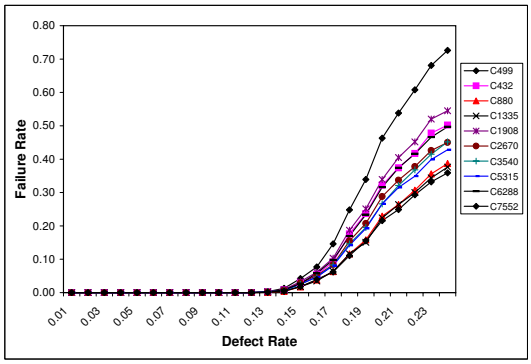

Fig. 5. Failure probability of synthesized ISCAS'85 benchmark circuits using Modified Tagged Repair technique.

TABLE I
Targeted defect rate of ISCAS'85 benchmark circuits synthesized into smaller $2^N \times N$ LUTs using Modified Tagged Repair Technique

	$2^2 \times 2$ LUTs	$2^3 \times 3$ LUTs	$2^4 \times 4$ LUTs	$2^5 \times 5$ LUTs	$2^6 \times 6$ LUTs	Targeted Defect Rate
C499	0	0	2	2	8	13.0%
C432	3	1	2	1	5	13.0%
C880	1	4	2	4	5	14.0%
C1335	5	0	4	5	5	13.0%
C1908	2	2	2	4	10	13.0%
C2670	2	4	5	3	9	13.0%
C3540	6	2	10	13	22	13.0%
C5315	8	8	12	8	25	13.0%
C6288	20	4	14	9	48	13.0%
C7552	6	18	16	20	30	14.0%

As seen from the results shown in Fig. 4 the modified technique further improves the defect tolerance of the given LUT architecture when compared to the Tagged Repair technique. Taking an example of a $2^3 \times 3$ LUT, we compare the results of Fig. 3 with Fig. 4. It can be seen that while the Tagged Repair technique can achieve 0% failure rate at defect rates of upto 17%, the modified technique can target defect rate upto 20%. This improvement in efficiency is due to the more optimized usage (by splitting the columns in two before applying repair) of the redundant spare units.

We also performed an analysis on the ISCAS'85 benchmark circuits in terms of targeted defect rate using Modified Tagged Repair technique. The failure rates for synthesized ISCAS'85 circuits are shown in Fig. 5. The ISCAS'85 benchmark circuits were first synthesized into smaller LUTs sizes (between $2^2 \times 2$ to $2^6 \times 6$). However, as can be seen from Table I, majority of synthesized circuits contain a higher proportion of $2^5 \times 5$ and $2^6 \times 6$ LUTs and hence the targeted defect rate of various benchmark circuits is around 13%-14% with little variation. This can be addressed by further synthesizing the circuits into LUTs of smaller sizes.

Fig. 6 compares the efficiency of the proposed techniques with the Repair Most technique [5] for LUTs of size $2^4 \times 4$ with 100% redundancy ($c_{sp} = 4$ spare columns and $r_{sp} = 2^4$ spare rows). As can be seen, the Modified Tagged Repair

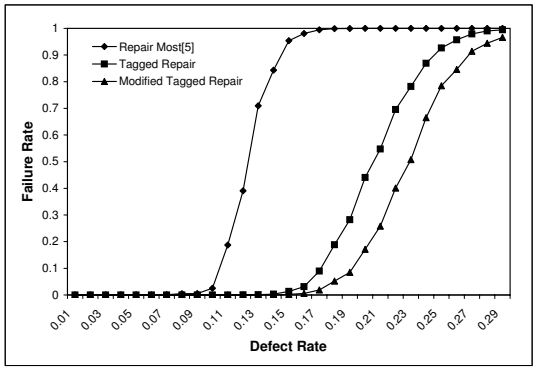

Fig. 6. Comparative study of the proposed repair techniques with the Repair Most technique for the failure rate of $2^4 \times 4$ LUT with 100% redundancy

TABLE II
COMPARATIVE REPAIR COST OF THE PROPOSED TECHNIQUES WITH THE REPAIR MOST TECHNIQUE IN TERMS OF TARGETED DEFECT RATE

		Targeted Defect Rate		
Spares	LUT size	Repair Most [5]	Tagged Repair (proposed)	Mod. Tagged Repair (proposed)
	3x3	7.0%	10.0%	12.0%
25%	4x4	4.0%	5.0%	6.0%
	6x6	2.0%	3.0%	4.0%
	3x3	8.0%	11.0%	14.0%
50%	4x4	6.0%	9.0%	9.0%
	6x6	2.0%	6.0%	6.0%
	3x3	9.0%	17.0%	20.0%
100%	4x4	8.0%	14.0%	15.0%
	6x6	2.0%	10.0%	11.0%

technique targets the highest defect rate followed by the Tagged Repair and Repair Most respectively. For example, when the defect rate is 15%, the Modified Tagged repair technique gives a failure rate of 0%, and the original Tagged Repair technique gives a failure rate of 2%, whereas, the Repair Most technique almost fails completely.

B. Repair Cost

To calculate the nanodevice area advantage of the proposed techniques, the total nanodevice area of $2^4 \times 4$ LUT implementation with 100% spares for the proposed techniques (Fig. 1 and 2) will be $3 \times 2^4 \times 4 = 192$ units/LUT. However, due to the implementation architecture of Repair Most technique, the total area (including original LUT and the spare units) of the Repair Most implementation will be $4 \times 2^4 \times 4 = 256$ units/LUT, which is 25% more as compared to the proposed techniques. The CMOS area overhead for Tagged Repair technique will be a total of $2 \times (2^4 + 4) = 40$ single bit tags. However, for the Modified Tagged Replacement technique, the CMOS area overhead will be $2 \times (2^4 + 4 + 4) = 48$ tags for a single LUT which will result in $8 \times 6 = 48$ extra CMOS transistors/LUT as compared to Tagged Repair technique (Fig. 1). Table II shows the comparative repair cost of the two proposed techniques with the Repair Most technique in terms of targeted defect rate. It can be seen that in case of a $2^3 \times 3$ LUT, Modified Tagged Repair can target upto 12% defect with only 25% spares, whereas Repair Most technique is not able to target 10% defect rate even with 100% spares. The targeted defect rate values given in this table have been rounded off to the nearest 1%. Similarly the number of spare units has been rounded off to the nearest whole number based on percentage of spares.

IV. CONCLUSION

In this work we proposed two efficient repair techniques for emerging technology (nano/CMOS) computational architecture implemented as LUTs. We have shown that these redundancy-repair techniques, while simple to implement, also have low redundancy requirements and are able to provide a high level of defect tolerance. We showed that while recently proposed Repair Most technique could handle defect rates of only upto 10% in case of $2^3 \times 3$ LUT our proposed Tagged Repair techniques showed much higher efficiency and were shown to handle defect rates of upto 20% for same size LUT. We also demonstrated the efficiency of our proposed techniques for larger circuits using synthesized ISCAS'85 benchmark circuits.

V. ACKNOWLEDGEMENT

The authors would like to acknowledge the EPSRC (UK) for funding this project in part under grant EP/E035965/1.

REFERENCES

[1] M. Ziegler and M. Stan, "CMOS/nano co-design for crossbar-based molecular electronic systems," *Nanotechnology, IEEE Transactions on*, vol. 2, pp. 217–230, Dec. 2003.

[2] C. Jeffery, A. Basagalar, and R. Figueiredo, "Dynamic sparing and error correction techniques for fault tolerance in nanoscale memory structures," *Nanotechnology, 2004. 4th IEEE Conference on*, pp. 168–170, Aug. 2004.

[3] F. Sun and T. Zhang, "Defect and Transient Fault-Tolerant System Design for Hybrid CMOS/Nanodevice Digital Memories," *Nanotech.*, vol. 6, no. 3, pp. 341–351, 2007.

[4] A. DeHon et.al., "Nonphotolithographic nanoscale memory density prospects," *Nanotechnology, IEEE Transactions on*, vol. 4, pp. 215–228, March 2005.

[5] D. B. Strukov and K. K. Likharev, "Prospects for terabit-scale nanoelectronic memories," *Nanotech.*, vol. 16, no. 1, pp. 137–148, 2005.

[6] M. Stan et.al., "Molecular Electronics: From Devices and Interconnect to Circuits and Architecture," *Proceedings of the IEEE*, vol. 91, pp. 1940–, Nov. 2003.

[7] W. Zhang et.al., "Nature: a hybrid nanotube/cmos dynamically reconfigurable architecture," in *DAC '06: Proceedings of the 43rd annual conference on Design automation*, (New York, NY, USA), pp. 711–716, ACM, 2006.

[8] D. Strukov and K. Likharev, "CMOL FPGA: a reconfigurable architecture for digital circuits with two-terminal nanodevices," *Nanotechnology*, vol. 16, no. 6, pp. 888–900, 2005.

[9] A. Singh et.al., "A heterogeneous CMOS-CNT architecture utilizing novel coding of boolean functions," *NANOARCH 07*, pp. 15–20, Oct. 2007.

[10] S. Paul, R. S. Chakraborty, and S. Bhunia, "Defect-aware configurable computing in nanoscale crossbar for improved yield," *IEEE International On-Line Testing Symposium*, vol. 0, pp. 29–36, 2007.

[11] S.Lu and C. Hsu, "Fault Tolerance Techniques for High Capacity RAM," *IEEE Transactions on Reliability*, vol. 55, pp. 293–304, June 2006.

[12] M. Mishra and S. Goldstein, "Defect tolerance at the end of the roadmap," *In ITC*, vol. 1, pp. 1201–1210, 30-Oct. 2, 2003.

[13] A. Bellaouar and M. Elmasry, "Low-Power Digital VLSI Design: Circuits and Systems," *Springer Publication*, . 1995.

[14] "http://www.synplicity.com/,"